64

GRAPHIC
SCREEN
EXPORTER

# UNIVERSAL GRAPHIC CONVERTER

## CONVERTS ANYTHING TO ANYTHING

| | |
|---|---|
| KOALA | KOALA |
| PAINT MAGIC | PAINT MAGIC |
| BLAZING PADDLES | BLAZING PADDLES |
| SORCERER'S APPRENTICE | SORCERER'S APPRENTICE |
| SCREEN MAGIC | SCREEN MAGIC |
| CAD GEM | CAD GEM |
| MOVIE MAKER | MOVIE MAKER |
| MICRON EYE | MICRON EYE |
| FLEXIDRAW | FLEXIDRAW |
| DOODLE | DOODLE |
| CBM PRINT SHOP | CBM PRINT SHOP |
| NON-CBM PRINT SHOP | NON-CBM PRINT SHOP |
| MICRO ILLUSTRATOR | MICRO ILLUSTRATOR |
| COMAL | COMAL |
| LOGO | LOGO |

UNIVERSAL

PROGRAM BY EVELYN ROSA

DESIGN BY WESLEY JAMES

# Graphic Screen Exporter

## PREFACE

When I first got my COMMODORE 64 the only available graphics software was SORCEROR'S APPRENTICE. Then came DOODLE, KOALA PAINTER, PAINT MAGIC and so on. In between the graphic programs where graphic extensions to Basic. Some were good, some where bad. The problem I ran into was how to get my good pictures out of the bad programs.

I got hold of a machine language monitor and went searching for my pictures in memory. At first it was difficult but, as time went on I came to be able to recognize pictures using the Interogate command in the monitor. Still, it was always a tedious job to find the load address, load the monitor, load the picture file, transfer parts of the file to where I needed them, add missing sections and save the files.

I decided to sit down and write a program to do all this. The first version of EXPORTER came into existence. It was called CONVERTER back then and the only thing it could do was convert pictures to DOODLE. That was sufficient at the time. As more Graphics packages came to market, however, I found myself back in the monitor.

Fortunately, by the time I was ready to write the next version I had learned some PDL. PDL, which stands for Program Developement Language is both a language and development procedure. With PDL and the help of it's creator, Wesley James, we were able to come up with the best algorithm and program structure possible. PDL also produced most, if not all, of the documentation that comes with the program. Source listings of the BASIC and ML routines used in the program were also produced during the "PDL'ing" of the program. All of this information is available to any registered owner who is interested. To obtain this information write to MIT/COMPUTER TOOLS DIV. requesting the additional documention and include a stamped self addressed envelope. The postage is 56 cents.

In any case, this program, in a sense, unites all the existing graphics programs and graphics extensions. It, therefore, effectively unites all of us. We at MIT/COMPUTERS TOOLS would like to here from you. Write (or call) and let us know what uses you have come up with for EXPORTER. Or, if you need information on how to get EXPORTER to do more, feel free to contact us. It is our hope that by providing this service we may be able to start a graphics oriented newsletter. This newsletter would contain tips and tricks from you, the users, and us as both developers and users on how to get the most out of the existing graphic packages in conjuntion with GRAPHIC SCREEN EXPORTER 64. I'll give you an inside scoop, if you let us know about other things you'd like a graphics package to do we may, be able to, include them in a system we're developing called THE DREAM GRAPHICS SYSTEM.

EVELYN ROSA
New York, NY, Sept. 1985

# Graphic Screen Exporter



MIT
MICRONIC
INTERFACE
TECHNOLOGIES

Computer Tools Div.

159 West 53rd St.
Suite 15B
New York, NY 10019

## CONTENTS

## GETTING STARTED

Place the EXPORTER disk in the drive. Make sure the disk drive door is in the closed position.

Type: LOAD"EXPORTER",8,1

When prompted place the disk which contains the file you wish to convert into the drive and press the space bar. The directory of the disk will be displayed 15 lines at a time.

If there are more than 15 entries press any key except the space bar. Press the space bar to make a selection.

Type in the name of the file and press <RETURN>, or press the <CTRL> key and 'C' to abort.

NOTE: For KOALA files a minimum of 7 characters must be typed.
EXAMPLE:"?PIC A "

## INCOMING FILES

If EXPORTER does not recognize the file type it will ask you which type it is. Use the cursor keys to move to your selection and then press <RETURN>.

## NEW FORMATS

EXPORTER will now ask you which file type you wish to convert to. Use the cursor keys to move to your selection and then press <RETURN>. The next prompt will ask you if you wish to erase the color and bit map information already in memory. If you are not sure simply hit the <RETURN> key.

Once the file has been loaded you can view it, alter it, change the colors, or flip back and forth between Hi-res and Multicolor modes. If the border is black you are in Hi-res Mode and if the border is white you are in Multicolor Mode

NOTE: Flipping from Hi-res to Multicolor and back will not effect the final file format.

## HI-RES TO MULTICOLOR

In the case of Hi-res to Multicolor conversions (ie: Doodle to Koala), there are two methods that can be used to optimize resolution.

1. If the picture was done in only two colors, page and pixel color, then two things need be done, shift and adjust. ADJUST will double all lit pixels that are not already paired. All '01' and '10' pixel combinations will become '11'. This means that in Multicolor Mode all pixels will get their color from color ram which is register #4. This leaves registers #2 and #3 for further color addition. Using the cursor keys to shift the bitmap left and/or right before you adjust will re-align the pixels. Experiment shifting and adjusting until you get the best picture.

2. If the Hi-res picture has several colors in it then these colors must be preserved. In order to do this you must transfer they color information to the proper areas of memory were Multicolor Mode can display them. This requires three steps;

1)<N> - PHOTO NEGATIVE the bitmap
2)<C> - transfer Hi-res pixel color to Multicolor #4 (color ram)
3)<D> - change all unset pixel pairs ('00') to register #2 ('01')

When you have done all three the colors will be in registers #2 (Hi-res background colors) and #4 (Hi-res pixel colors). This leaves you with registers #1 and #3 to add any additional colors without conflicting with existing ones

## BRINGING IN OTHER FILE TYPES

FLEXIDRAW & LOGO SCREENS: Bring in the bitmaps as PS-Screen Magic files and if there are color files abort the save by pressing <CTRL-C> and bring in the color files as if they were Doodle files. DO NOT ERASE color and bitmap memory.

FONTS: Bring in any font that is in the same format as the normal CBM character set as a PS-Screen Magic file.

COMPUTEREYES: Standard Files may be brought in as PS-Screen Magic files.

## UNIVERSAL

To convert picture files that are not in the same format as any of the ones provided, choose UNIVERSAL as the format

When you choose UNIVERSAL, EXPORTER will ask you a series of questions about the contents of the picture file. Be prepared to answere the following questions:

1.Is the picture MULTICOLOR or HI-RES?

2.How many bytes long are each of the sections that make up the picture file?

-Bitmap, Screen Ram, Color Ram, and Extra bytes

BITMAP: This section can be no more than 8192 bytes long. If you enter a length less than 8000 you will be asked whether the missing bytes are at the begining or at the end of the bitmap.

SCREEN RAM and COLOR RAM: These sections can be no more than 1024 bytes each. If you specify that they are only one (1) byte long then you must provide their location in the file. In other words, how many bytes from the beginning (first byte) of the file is this byte. In doing this calculation disregard the load address of the file if any.

EXTRA BYTES: These are gap bytes, if any, that a file might have seperating any of the other sections.

3.In what order are the sections of the file?

There are six possible sections to a file. These are:

| | |
|---|---|
| BITMAP | *SCREEN COLOR |
| SCREEN RAM | BORDER COLOR |
| *COLOR RAM | EXTRA BYTES |

* MULTICOLOR only.

4. Is this a PROGRAM or a SEQUENTIAL disk file?

5. PROGRAM files only. What load address should this file have. Use 'FIND ADDR' to get this value from any existing PROGRAM file.

Using this information EXPORTER will calculate seven addresses which will be used through out the program. It uses them when it

brings in the file, when it arranges the file for display and when it assembles a file to save on the disk. For more information concerning these seven addresses load "-CREATE ADDR" and run it. The instructions will give you a description of each address.

## SELF IMPLEMENTING UNIVERSAL

If there is a format that you will be needing frequently, you can bypass all of the questions asked by UNIVERSAL by altering the program itself.

To do this follow the step by step instructions listed below. NOTE: This method is for PROGRAM files only.

1. Make a backup of the Exporter disk using any file or disk backup program.

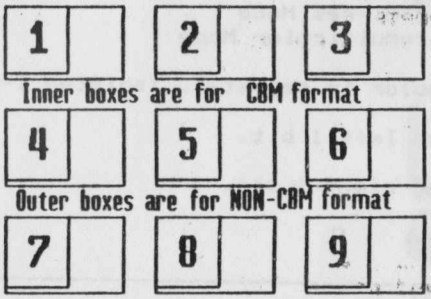
2. Load and run EXPORTER
3. Give a name when asked for the file name.
4. Choose Universal as the incoming file type.
5. Answer all questions concerning the file.
6. When asked 'Convert To What Format' press the RUN/STOP key and then move your cursor to the bottom of the screen.
7. Type in the following line without a line number:
   FOR I=0 TO 13:PRINT A(12,I):NEXT <RETURN>
8. The stream of numbers that will be printed are the seven addresses in order. Write these 14 numbers down!
9. Type 1260 FT=0 and press <RETURN>
   Type 1330 FT=0 and press <RETURN>
10. Save your altered version of EXPORTER to your backup disk as follows:
    SAVE"@0:EX.PROG",8 <RETURN>
11. Load "-CREATE ADDR",8
12. Type LIST 340-350 and press <RETURN>
14. In place of the 14 zeros in line 350 type in the 14 numbers that where printed using instruction #7. Remember to place commas in between the numbers and don't forget to press the <RETURN> key after making the changes.
15. Save your altered version of "-CREATE ADDR" on your backup disk.
16. Run your altered Version of "-CREATE ADDR".

Now when you run your altered backup of EXPORTER and choose UNIVERSAL it will no longer ask you any of the questions. It will have all the information that it needs already.

## PRINT SHOP

This format when chosen will allow conversion of CBM and NON-CBM PRINT SHOP graphic stamps. You can even convert CBM to NON-CBM and vice versa.

When you choose this format you will be asked which of the nine available slots you wish the stamp brought into or taken out of. Included on the EXPORTER disk is PS.TEMPLATE which is a PS-SCREEN MAGIC/FLEXIDRAW compatible file which contains the layout of the nine slots. This screen can be used to position the stamps that you wish to create.
Below is an example of the PS.TEMPLATE screen. The boxes are positioned so that their left sides are on the start of an 8 by 8 character column on the screen.




NOTE: When converting stamps to FLEXIDRAW all nine CBM slots may be used, but only the first two columns of NON-CBM slots may be used since FLEXIDRAW allows use of only a 264 by 200 HI-RES screen instead of the normal 320 by 200 screen.

Those who have PAINT MAGIC can convert the template screen and a screen which contains what will become stamps into PAINT MAGIC format. You can then use the MERGE feature in PAINT MAGIC to place your graphics within the positioning boxes on the template screen.

When converting to PAINT MAGIC do not use the ALTER ,NEGATIVE or OR options as this will change the appearence of your final stamp.

## COMMANDS IN VIEW MODE

| Key | Effect |
|---|---|
| A | ADJUST bitmap from Hi-res to Multicolor. (pairs all lit pixels) |
| C | Move Screen Ram to Color Ram (color in reg.3 to reg.4) |
| H | Return to help screen. |
| N | PHOTO NEGATIVE of bitmap |
| O | Change bit pairs for color in register 1 to register 2. |
| R | RESTORE original bitmap and colors. |
| F1 | Toggle Hi-res and Multicolor Modes.<br>Black border=Hi-res Mode<br>White border=Multicolor Mode |
| 1-4 | Increment color in register. (shifted 1-4 decrement color) |
| CrsrLFT | Shift bitmap left 1 bit. |
| CrsrRT | Shift bitmap right 1 bit. |
| RETURN | Save bitmap. |

## HELFUL HINTS

To convert other file types not listed in the menu's follow these steps:

1) Use 'ADDR FINDER' to get the load address of the file.
2) Use the monitor to find out what order the bitmap and color information are in.
3) Calculate how many bytes each section takes up.
4) Choose the file format(s) most similar to the one(s) you need.
5) Make any changes in the load address, file name or file type using 'DISK EDITOR' and 'DOS ENABLE'.

# APPENDIX

## MACHINE LANGUAGE MONITORS

LOAD MONITORS AS FOLLOWS:

Type: LOAD"MON$C",8,1 and press <RETURN>
then type SYS49152 <RETURN>

or

Type: LOAD"MON$8",8,1 and press <RETURN>
then type SYS32768 <RETURN>

## DOS ENABLE

To activate DOS ENABLE commands;

Type: LOAD"BOOT DOS",8,1 and press <RETURN>

## DISK EDITOR

Activate disk editor as follows:

Type: LOAD"DISK EDITOR",8 and press <RETURN>

Once it has been loaded, insert disk to be edited:

Type: RUN and press <RETURN>

DISK EDITOR has four modes of operation they are: BAM mode, Directory mode, File mode, Edit mode

BAM MODE: If you enter Track 18, Sector 0 (18,0) and answer N to the question, "Do you wish to Edit?" you will be in BAM mode. In BAM mode the map is a graphic representation of the block usage of the disk. Solid yellow areas represent blocks that have not been allocated (unused). The black dots indicate allocated blocks (containing information).

DIRECTORY MODE: Track 18 and any valid sector other then 0 and then N to the Edit question takes you to Directory mode.
Regarding Directory mode the following should explain all:

```
18  4   (this is the next sector of the directory in decimal
         block 0 255 means there are no more)

82      17 00 FILE NAME  9   0
```

The 82 above is the file type (hex). See DOS FILE TYPES table.
The 17 00 are the track and sector (decimal) in which the file starts.
The file name is self explanatory.
The numbers 9 0 are the size of the file in low byte, high byte order (decimal).

FILE MODE: Any track other than 18 and N to the Edit question puts you in File mode. It displays your selected track and sector as hex values.

EDIT MODE: Answer Y to the Edit question. Once you are in Edit mode you may move the cursor any where on the screen and over-write any character. DO NOT write in any of the blank spaces.

Press <F1> to rewrite the current track and sector back onto the disk

HARD COPY: From any of the modes it is possible to get a printout of the screen. When the prompt PRESS ANY KEY appears, type <CTRL-P>, you'll get a direct screen dump.

To quit enter 0,0 as the track and sector.

## SYSTEM MEMORY MAPS

| HEX | DECIMAL | CONTENTS |
|---|---|---|
| 0334-0335 | 820-821 | File length |
| 033D | 829 | Filename length |
| 033E-0355 | 830-853 | Filename |
| 0356-035B | 854-859 | Machine language routines |
| 035C-035D | 860-861 | Screen color (reg. #1)/border color |
| 035E-035F | 862-863 | Color for reg. #3 and #4 |
| 0360-0365 | 864-869 | Machine Language Routine |
| 0366-03FB | 870-1019 | Not Uesed |
| 0800-54FF | 2048-21756 | Exporter program space |
| 5500-9BFF | 21760-39935 | Exporter WORK SPACE |
| 5C00-5FFF | 23552-24575 | Screen Ram (display) |
| 6000-7FFF | 24576-32767 | Bitmap (display) |
| 8000-8FFF | 32768-36863 | Workspace |
| 9000-93FF | 36864-37887 | Original Color Ram for Restore |
| 9400-97FF | 37888-38911 | Original Screen Ram for Restore |
| 9800-9BFF | 38912-39935 | Not used |
| 9C00-9FFF | 39936-40959 | Machine Language routines (* see below) |
| A000-BFFF | 40960-49151 | Extra Bitmap for Restore/BASIC ROM |
| C000-CFFF | 49152-53247 | Not used. 'MON$C' or 'DOS ENABLE' |
| D000-D7FF | 53240-55295 | I/O BLOCK |
| D800-DBFF | 55296-56320 | Color Ram Nybbles |

(*)-The following chart shows the requirements for using the ML routines located from $9C00 - $9FFF

## ML ROUTINE REQUIREMENTS

| NAME | GO | SYS | REQUIRES |
|---|---|---|---|
| ADJUST | $9E18 | 40472..... | Bitmap at $6000 (24576) |
| DIRECTORY | $9C0C | 39948..... | OPEN2,8,0,"$" before calling |
| EX/OR | $9F31 | 40732..... | Bitmap at $6000<br>Exclusive OR :value<br>POKE 40762,73:POKE40763,value<br>Logical OR :value<br>POKE 40762,9 :POKE 40763,value |

## ML ROUTINE REQUIREMENTS

```
NAME        GO      SYS          REQUIRES
-----------------------------------------------------------------------
FILL        $9C06   39942.......From-$FD, Length-$0334, Value-$FB
                                   253        820          251

LEFT        $9E1B   40475.......Bitmap at $6000
SHIFT

LOAD        $9C09   39945.......To-$FB, Name-$033E, Name lngth-$0334
                                 251      830            820

MEMORY      $9C03   39939.......From-$FD, To-$FB, Length-$0334
MOVE                               253      251        820

NIBBLE      $9F2B   40732.......From-$FD, VALUE-$FB
FILL                               253      251
                            Flag-$FC (15=high, 240=low)
                                   252

PHOTO       $9F31   40753.......Bitmap at $6000
NEGATIVE                         POKE 40762,73:POKE 40763,255

RESTORE     $9E15   40469.......Original bitmap at $A000
                                 (restores bitmap only)

RIGHT       $9E1B   40475.......Bitmap at $6000
SHIFT

SAVE        $9C00   39936.......From-$C1, To-$AE, Name-$33E-$34E
                                 193      174      830-846
                            Name lngth-$033E, Load Addr-$FB
                                     830              251
-----------------------------------------------------------------------
 NOTE: To, From & Length are two byte values in low byte,
       high byte order.
```

## MONITOR COMMANDS

| CONMMAND | FORMAT | DESCRIPTION |
|---|---|---|
| ASSEMBLE | .A n1 opcode operand<br>.A 033C LDA #$FF | Enter a line of assembly code at location n1 |
| COMPARE MEMORY | .C n1 n2 n3<br>.C 2000 2100 5000 | Compare memory from n1 to n2 with memory at n3 and report differences |
| DISASSEMBLE | .D n1 (n2)<br>.D 2000 2100 | Disassemble code starting at n1. Continue to (n2) |
| FILL | .F n1 n2 value<br>.F 2000 2100 00 | Place 'value' in all locations from n1 to n2. |
| GO | .G n1<br>.G 033C | Execute code starting at n1. |
| HUNT | .H n1 n2 data<br>.H 2000 2100 'text<br>.H 2000 2100 0F 4C | Search memory from n1 to n2 for data & report their addresses. |
| INTEROGATE | .I n1 (n2)<br>.I 2000 2100 | Display memory from n1 (to n2) as ascii and hex |
| LOAD | .L"name",D#<br>.L"DDART",08 | Load a file from device, D# at the address specified by the file. |
| MEMORY | .M n1 (n2)<br>.M 2000 2100 | Display hex values in address n1 (thru n2). |
| NEW LOCATOR | .N n1 n2 offset<br>.N 2000 2100 0400 | Relocate absolute memory references from n1 to n2 by adding an offset |
| REGISTER | .R | PC = Program counter<br>SR = Status register<br>AC = Accumulator<br>XR/YR = X and Y registers<br>SP = Stack pointer |
| SAVE | .S"name",D#,n1,n2+1<br>.S"TEST",08,033C,0400 | Save n1 thru n2 to device D# using "name" |
| TRANSFER | .T n1 n2 n3<br>.T 6000 8000 4000 | Move memory at n1 thru n2 to n3 - |

## DOS FILE TYPES

| Value | High Nibble | State | | Low Nibble | Type | |
|---|---|---|---|---|---|---|
| $00-04 | $0 | Unclosed | * | $0 | Deleted | DEL |
| $80-84 | $8 | Normal | | $1 | Sequential | SEQ |
| $A0-A4 | $A | @ Replacement | | $2 | Program | PRG |
| $C0-C4 | $C | Locked | < | $3 | User | USR |
| | | | | $4 | Relative | REL |

## DOS ENABLE COMMANDS

| Format | Description |
|---|---|
| @ | Display disk status |
| @C0:newname=oldname | Copy or concatenate disk file(s) |
| @I0: | Initialize drive 0 |
| @N0:name(,id) | Rename or format a disk |
| @Q | Kill all DOS ENABLE commands |
| @R0:newname=oldname | Rename a file |
| @S0:name | Scratch a file |
| @S0:n1,n2,n3,n4 | Scratch up to 4 files at once |
| @$ | List directory to screen |
| @$0:c1* | List entries starting with the letter c1 |
| @#n | Send all DOS commands to drive #n where n is 8 to 15 inclusive |
| /name | Load file 'name' (at BASIC) |
| %name | Load file 'name' (at its load addr) |
| ↑name | Load and run file 'name' (at BASIC) |
| ←name | Save a file to disk |

# Graphic Screen Exporter

## HI-RES FORMATS

| Type (# blocks) | Load Addr. | # bytes | Content (in order) |
|---|---|---|---|
| PS-Screen Magic & Flexidraw (32) | $6000<br>24576 | 8192 | bitmap only. |
| Cad Gem (32) | $6002<br>24578 | 7998 | bitmap only. |
| Micron Eye (32) | $6280<br>25216 | 7360 | bitmap only. |
| Doodle (37) | $5C00<br>23552 | 1024<br>8192 | screen ram,<br>bitmap. |
| Print Shop (3),(2) | $5800<br>22528 | 572<br>288 | bit stream (non-CBM only)<br>bit stream (CBM only) |
| COMAL Hi-res (36) | $5C17<br>23575 | 1<br>1<br>1<br>1000<br>8000 | flag 0=hires<br>screen color<br>border color<br>screen ram<br>bitmap |

## ADDR FILE FORMAT

| Byte | Content (Parameters for all conversions) |
|---|---|
| 0,1 | Address used by EXPORTER'S load and save routines. |
| 2,3 | Start of Screen Memory |
| 4,5 | Start of Color Memory |
| 6,7 | Load address file needs |
| 8,9 | Location of last byte in file |
| 10,11 | Location of Screen color byte |
| 12,13 | Location of Border color byte |

NOTE:-All values are two bytes in low byte, high byte order.
-A value of 00 00 indicates the file does not contain or require this information

## MULTICOLOR FORMATS

| Type (# blocks) | Load Addr | # bytes | Content (in order) |
|---|---|---|---|
| Koala & | $6000 | 8000 | bitmap, |
| Super sketch | 24576 | 1000 | screen ram, reg. #2, #3 |
| (40) | | 1000 | color ram, reg. #4 |
| | | 1 | screen color. reg. #1 |
| Paint Magic | $5F8E | 114 | ml routine, |
| (37) | 24462 | 8000 | bitmap, |
| | | 192 | ml & color for reg. #4 |
| | | 1000 | screen ram, reg. #2, #3 |
| Blazing Paddles | $6000 | 8192 | bitmap, |
| & Sorceror's | 24576 | 1024 | screen ram, |
| Apprentice | | 1024 | color ram, |
| (41) | | 1 | screen color. |
| Movie Maker | $6000 | 7680 | bitmap, |
| (31) | 24576 | 4 | color/reg. #1, #2, #3, #4 |
| | | 1 | border color |
| COMAL Multicolor | $582F | 1 | Flag 1=multi |
| (40) | 22575 | 1 | screen color |
| | | 1 | border color |
| | | 1000 | screen ram |
| | | 1000 | color ram |
| | | 8000 | bitmap |
| Micro Illustrator | $581C | 20 | extra bytes |
| | 22556 | 1000 | screen ram |
| | | 1000 | color ram |
| | | 8000 | bitmap |

## HELFUL HINTS

The number of blocks a picture file takes up on the disk will usually tell you quite a bit about the file's contents.

-If it is 32 blocks or less then it is bitmap information only.
-If it is 36 or 37 blocks then it contains bitmap and color information, but not necessarily in that order. Also, it is most probably a Hi-res picture
-A file 40 blocks long is probably Multicolor, contains 1000 bytes each for color and screen memory and contains 8000 byte of bitmap information
-A file 41 blocks long probably Multicolor, contains 1024 bytes each for color and screen memory and contains 8192 byte of bitmap information